**Integrated Multi-Modal Spectral Monitoring System for Optimising Agriphotovoltaics and Polytunnel Agriculture**

**Executive Summary**

The "Integrated Multi-Modal Spectral Monitoring System for Optimising Agriphotovoltaics and Polytunnel Agriculture" is a low-cost, open-source IoT platform designed to enhance the synergy between sustainable energy production and agricultural productivity. Its core purpose is to address critical gaps in existing monitoring solutions for Agriphotovoltaics (AgriPV) and polytunnel environments.

**Why is this System Essential?** AgriPV and polytunnel agriculture demand precise, real-time monitoring of both environmental and PV parameters. Traditional systems are often expensive, proprietary, and lack the detailed, synchronised data needed to manage the intricate relationship between PV efficiency and plant health, especially under dynamically changing light conditions. This platform provides a unified, modular solution for comprehensive data capture.

**What does it do?** This system integrates full-spectrum spectral analysis, environmental sensing (temperature, humidity, pressure, gas), and PV performance monitoring (voltage, current, power) into a single, cohesive unit. It features an intelligent MPPT algorithm that adapts to varying light conditions, pausing during low light to prevent panel damage and resetting upon daybreak. All collected data is robustly logged to an SD card, making it ideal for remote or off-grid applications without continuous internet access. An integrated OLED display provides real-time feedback.

**Where is its Impact?** This platform is particularly impactful in the **Global South**, where high solar irradiance, climate volatility, and resource constraints hinder AgriPV adoption. By providing affordable, context-appropriate monitoring, it supports local decision-making for irrigation, shading, and crop selection. It also fosters climate-smart agriculture, education, and workforce development in sustainable practices.

**The Ultimate Aim: ML & DL Datasets!** A key objective of this project is to generate rich, integrated datasets specifically suitable for **Machine Learning (ML) and Deep Learning (DL) applications**. This multi-modal, temporally aligned data is invaluable for training advanced AI models to predict PV degradation, detect early signs of crop stress, and forecast energy output. The next phase of this project will focus on the development and application of these ML and DL models – watch this space!

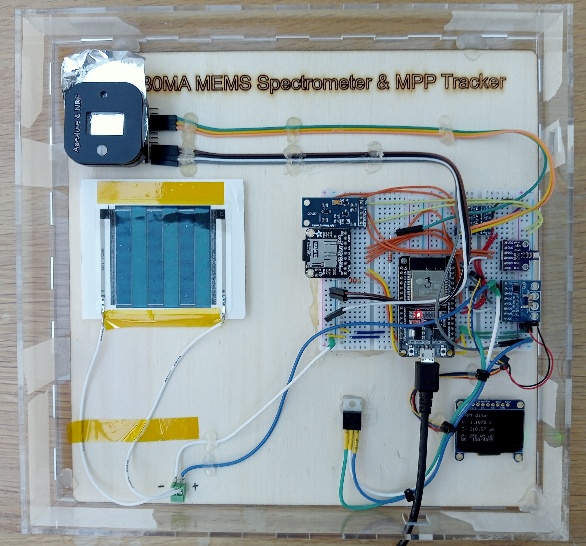
**1. System Implementation**

This section details the hardware, software, and deployment aspects of the Integrated Multi-Modal Spectral Monitoring System.

**1.1 System Design**

The system is built around an **ESP32 microcontroller**, selected for its low power consumption, integrated wireless connectivity, and GPIO flexibility. Key components of the system include:

* **Spectral Sensor**: A full-spectrum spectrometer (C12880MA) for capturing ambient light spectra.
* **Environmental Sensor**: A BME680 sensor for measuring temperature, humidity, and pressure.
* **Light Intensity Sensor**: A GY-30 (BH1750) for irradiance monitoring (lux).
* **PV Monitoring**: An INA226 power sensor with an MPPT-based tracking system for OpV, GaAs, and a-Si modules.
* **Storage**: An SD card module for synchronised data logging.
* **Display**: A 0.96" 128x64 OLED display (SSD1306) for real-time feedback.

****

**Figure 1: Complete System Overview**

**1.2 Sensor Integration and Calibration**

All sensors are integrated using I2C and analog interfaces. Calibration is performed using standardised light sources and reference measurements to ensure data accuracy.

**1.3 Deployment Protocol**

A prototype of the system was deployed in a controlled polytunnel. Data was logged every 30 seconds over a 7-day period, capturing diurnal cycles and varying environmental conditions.

**2. Hardware Requirements**

* ESP32 Development Board (e.g., ESP32-DevKitC)
* C12880MA MEMS Spectrometer Module
* INA226 Current/Voltage Sensor
* BME680 Environmental Sensor
* GY-30 (BH1750) Illuminance Sensor
* DFRobot MOSFET Power Controller Module (or similar PWM-controlled load)
* OpV (Organic Photovoltaic) Panel
* Micro SD Card Module
* Micro SD Card
* 0.96" 128x64 OLED Display (SSD1306, I2C)
* Breadboard and Jumper Wires
* 5V Power Supply

**3. Wiring**

* **ESP32 I2C**:
  + SDA: GPIO 21 (connected to SDA of INA226, BME680, GY-30, OLED)
  + SCL: GPIO 22 (connected to SCL of INA226, BME680, GY-30, OLED)
* **PWM for MOSFET**:
  + PWM Signal: GPIO 15
* **SD Card Module (SPI)**:
  + CS (Chip Select): GPIO 5
  + MOSI: Connect to ESP32 MOSI pin (typically GPIO 23)
  + MISO: Connect to ESP32 MISO pin (typically GPIO 19)
  + SCK: Connect to ESP32 SCK pin (typically GPIO 18)
* **C12880MA Spectrometer**:
  + CLK: GPIO 16
  + ST: GPIO 17
  + VIDEO: GPIO 34 (Analog input)
  + EOS: GPIO 25 (Input, optional for sync)
  + WHITE\_LED: GPIO 26
  + LASER\_404: GPIO 27

**Important**: Ensure all GND pins (ESP32, INA226, DFRobot MOSFET, OpV Negative, Spectrometer, BME680, GY-30, SD card, OLED) are connected to a common ground.

**4. Software Setup**

**4.1 Arduino IDE Setup**

* Install the Arduino IDE.
* Add ESP32 Board Manager URL: https://raw.githubusercontent.com/espressif/arduino-esp32/gh-pages/package\_esp32\_index.json
  + Go to File > Preferences.
  + Paste the URL into "Additional Boards Manager URLs".
* Install ESP32 Boards:
  + Go to Tools > Board > Boards Manager.
  + Search for "esp32" and install the "esp32 by Espressif Systems" package.
* Select your ESP32 board (Tools > Board > ESP32 Arduino > ESP32 Dev Module).
* Select the correct COM Port (Tools > Port).

**4.2 Library Installation**

Install the following libraries via Sketch > Include Library > Manage Libraries...:

* **INA226\_WE**: Search for "INA226\_WE" by Wolfgang Ewald.
* **Adafruit GFX Library**: Search for "Adafruit GFX Library" by Adafruit.
* **Adafruit SSD1306**: Search for "Adafruit SSD1306" by Adafruit.
* **Adafruit BME680 Library**: Search for "Adafruit BME680 Library" by Adafruit.
* **Adafruit Unified Sensor**: This is a dependency for BME680, install if prompted or search for it.

The Wire.h, SPI.h, SD.h, WiFi.h, and time.h libraries are typically included with the ESP32 core.

**4.3 WiFi Configuration**

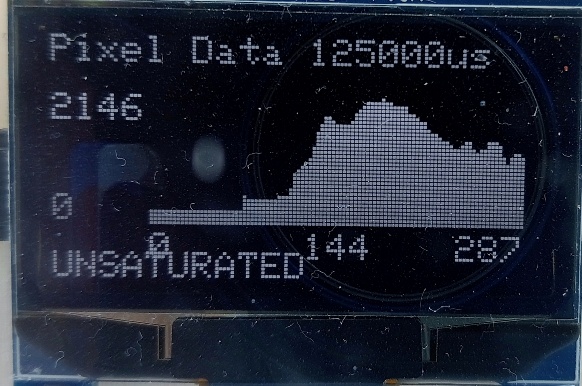
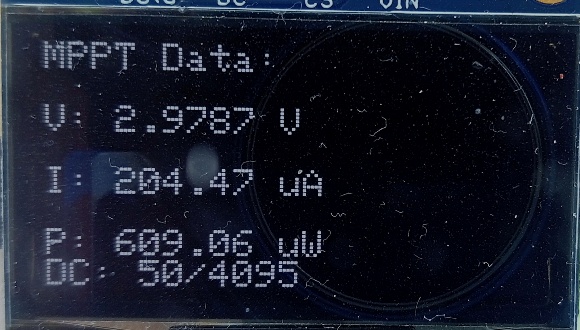
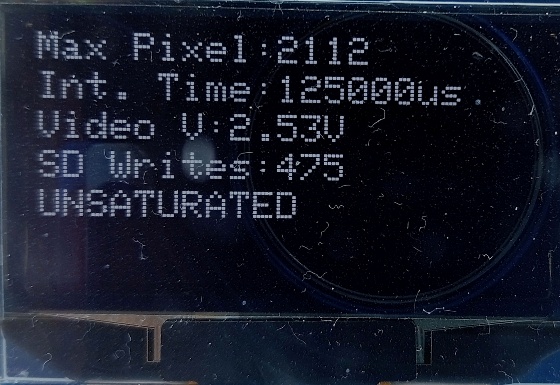
Update your WiFi credentials in the sketch:

const char\* ssid = "YOUR\_WIFI\_SSID"; // Replace with your WiFi SSID

const char\* password = "YOUR\_WIFI\_PASSWORD"; // Replace with your WiFi Password

**5. Usage**

1. **Upload the Sketch**: Open the ESP32\_C12880MA\_MPPT\_SD4.ino file in the Arduino IDE and upload it to your ESP32 board.
2. **Monitor Serial Output**: Open the Serial Monitor (Baud Rate: 115200) to observe initialisation messages, sensor readings, MPPT status, and SD card write confirmations.
3. **OLED Display**: The OLED will cycle through different data pages: Spectrometer stats, Spectrograph plot, MPPT data, and Environmental sensor readings.

****

**Figure 2: Example OLED Display Output**

1. **SD Card Logging**: Data will be logged to SENSOR\_LOG.CSV on the SD card when the ambient light (Lux) is above the LUX\_MPPT\_THRESHOLD (50 lux).
2. **MPPT Operation**: The MPPT algorithm will actively track the maximum power point during sufficient light conditions and park the MOSFET during low light.

**6. Customisation**

* **MPPT Parameters**: Adjust DUTY\_CYCLE\_STEP, MPPT\_INTERVAL\_MS, LUX\_MPPT\_THRESHOLD, and DUTY\_PARK constants to fine-tune the MPPT algorithm for your specific OpV panel and load.
* **Spectrometer Gain**: Modify TARGET\_MAX\_HIGH, TARGET\_MAX\_LOW, MIN\_INTEGRATION\_TIME\_US, and MAX\_INTEGRATION\_TIME\_US for optimal spectrometer performance.
* **SD Card Logging Interval**: Change the 10000 (10 seconds) in the lastSensorDataSaveTime condition within loop() to alter the logging frequency.
* **NTP Settings**: Adjust gmtOffset\_sec and daylightOffset\_sec for your local timezone if BST is not applicable.

**7. Troubleshooting**

* **I2C Devices Not Found**: Run the I2C scanner. Check wiring for SDA/SCL.
* **INA226 Error / BME680 Error / GY-30 Error**: Re-check sensor wiring and power. Ensure correct I2C addresses (default addresses are used in the code).
* **SD Card Initialisation Failed**: Check wiring for CS, MOSI, MISO, SCK. Ensure the SD card is properly formatted (FAT32) and inserted.
* **WiFi Connection Failed**: Double-check ssid and password. Ensure your ESP32 is within range of your WiFi network.
* **NTP Sync Failed**: Verify WiFi connection. Ensure pool.ntp.org is reachable from your network.
* **PWM Issues**: Ensure PWM\_PIN is correctly connected to the MOSFET module. Verify PWM\_FREQUENCY and PWM\_RESOLUTION\_BITS are appropriate for your MOSFET driver.

**Copyright**

© 2025 K. Seunarine. All rights reserved.

**License**

This project is open-source and released under the [MIT License](https://opensource.org/licenses/MIT).